2020

Detection of curbside storm drain from street level images using Faster R-CNN

Prabhakaran Pichaikutty
Iowa State University

Detection of curbside storm drains from street-level images using Faster R-CNN

by

Prabhakaran Pichaikutty

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Electrical Engineering (Systems and Controls)

Program of Study Committee:
Dr.Joshua Peschel, Major Professor
Dr.Hongwei Zhang
Dr.Matthew Darr

The student author, whose presentation of the scholarship herein was approved by the program of the study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

ACKNOWLEDGMENTS

I would like to sincerely express my thanks to Dr.Joshua Peschel (Department of Agriculture and Biosystems Engineering) for his constant guidance and for providing me the opportunity to work on this research. Without his support, this work would not have been possible. I would also like to thank Dr.Hongwei Zhang (Department of Electrical and Computer Engineering) and Dr. Matthew Darr (Department of Agriculture and Biosystems Engineering) to be a part of my thesis committee. I would like to thank my research group members and a special thanks to Abdullah Souvrav for their time and advice.

I am grateful for my friends at Iowa State University, whose encouragement and constant support have contributed significantly to my efforts on completing the thesis and my overall experience at Ames, Iowa.

Finally, I am deeply grateful to my family for their patience and constant support throughout my entire graduate school period.

ABSTRACT

Stormwater management is a significant part of modern urban infrastructure. With an increase in climate change due to global warming, this system plays a major role in conserving water and maintaining the environment. Also, they play a significant role in risk management in times of flood. Storm drains/inlets are essential in modeling this system. Precise mapping of the location of these drains is the key step in improving the infrastructure. State of the art deep learning technique using Faster R-CNN is presented in this thesis to identify the drains on the curb of the road using Google street view images as the primary source. The model is evaluated with 1000 street-level images of streets and highways of Urbana-Champaign, Illinois-USA. The method proposed shows a significant improvement in the detection accuracy of drains by eliminating a significant amount of false positives compared to the previous state of the art machine vision detection techniques. The dataset used for the thesis is available for future researchers.

CHAPTER 1. INTRODUCTION

Stormwater is rainwater or melted snow that runs off streets, lawns, and other sites. When stormwater is absorbed into the soil, it is filtered and ultimately replenishes aquifers or flows into streams and rivers [1]. Stormwater management systems are designed to collect the runoff water through an underground transference system to a location where it is treated or directly returned to rivers or reservoirs. A working stormwater management system is necessary to prevent flooding of homes and businesses. This system is crucial for public health and environmental health since it is also used to transport sewage. Sewage pollutes the water with pathogens, heavy metals, and other toxins, causing serious health problems. They can also cause algal bloom due to excess nutrients and kills aquatic life. Storm Drains are the access points to this critical system. Modeling the current infrastructure plays a significant role in understanding and improving the stormwater management system.

Regardless of the drains' significant role in the stormwater management system, their whereabouts and condition are commonly obscure. Documentation and review of these systems are established through an audit conducted by a municipality; Which is a hugely time-consuming process, and the accuracy of the results greatly varies. The inaccurate data will significantly impact the understanding of the system as a whole. Depwe [2](2015) developed a tool to extract the storm drain's location for the desired locality. The tool has a java based GUI to download the Google Street View images for the given co-ordinates. A computer vision-based processing algorithm is run on the downloaded images to detect the drains. The tool provides a CSV file containing the latitude and longitude information as an output with data exported from the processing algorithm. The main issue with that tool is the accuracy of the processing algorithm, which is greatly affected due to environmental features in the image and limitations of the

computer vision algorithm to discard those features. This study's primary objective is to develop a new robust processing algorithm to detect the drains with improved accuracy and reduce false positive detection compared to previous geometrical machine vision techniques. In this study, a deep-learning approach is used for the processing algorithm, which uses the Faster R-CNN object detection algorithm. The following section reviews previous studies which use Faster R-CNN for object detection problem.

Related Works

Street view images are used in various deep learning applications as the primary source of assessment data. GSV images are the most common source of street view images, which are free and can be accessed through a simple HTTP request. Compared to satellite imagery,street-level imagery describes the urban environment with close details. Research in recent years shows that street view images are a reliable data source for many applications, which includes but not limited to urban social sensing (Zhang et al. 2019 [3]), Demographic estimation (Gebru et al. 2017 [4]), Traffic sign detection (Lu et al. 2018 [5]), Urban environment (Rundle et al. 2011 [6]).

Storm drains were detected using various computer vision approaches. Depwe [2] created a computer vision algorithm to detect grated drains from street-level images. Pasquet et al. [7]developed an algorithm to detect manholes from high-resolution aerial images. The algorithm combines circular object detection using computer vision and a machine learning approach to build a detection model. They reported an accuracy of 40% with a precision of 80%. In both cases, the accuracy of the algorithm is affected by lighting conditions, noise, and other environmental conditions. Traditional machine vision and machine learning techniques have a low probability of detecting storm drain due to non-linear features. Deep-learning based methods have a better performance in extracting features as well as detecting objects.

Deep learning techniques are used in various classification and detection applications across various fields. The use of the street level imagery as the primary data source for deep learning model in urban infrastructure is gaining popularity. Zhang et al. (2016) used a deep convolutional neural network to detect cracks on roadways from images captured through a smartphone. The neural network is trained using SGD and a batch size of 48 images. The proposed model is compared with SVM and Boosting techniques. Based on the reported results, the deep learning model outperformed the normal machine learning techniques.

Campbell et al. [8] used GSV images to detect traffic signs using an SSD mobile object detection model. The model is reported to have an accuracy of 95.63% and stated that the deep learning approach on detecting traffic signs outperformed the previous models, which used computer vision techniques and machine learning algorithms.

CNN-Based Object Detection

Various CNN based algorithms are proposed for object detection. Girshick et al. [9] proposed the region-based CNN called RCNN, which combines the regional proposal algorithm and CNN as a feature extractor. Instead of working on the entire image, the algorithm extracts regions through selective search. To avoid redundant computations on the areas, the same author improved the algorithm by adding an ROI pooling layer and a pyramid spatial pooling layer (Fast R-CNN [10]). The input image is passed to the convolutional layers instead of the regions, and the regions of interest are extracted from the convolutional feature map, which improved the training and testing times significantly. Faster R-CNN proposed by Ren et al. [11] replaces the selective search with the Regional proposal network, which shares the same convolutional layer with the classification network. Faster R-CNN achieves the state of the art results on PASCAL VOC 2007, 2012, and MS COCO datasets with fewer regional proposals than its predecessors.

YOLO (you only look once) proposed by Joseph Redmon is a single-shot technique wherein the nxn image is passed to the fully convolutional layer only once and conducts bounding box regression. SSD is another single-shot approach where the ROI pooling and region proposal production are left out. This makes these single-shot algorithms faster in detection, but the accuracy is less than fast R-CNN and Faster R-CNN.

All available models recorded a mean average precision score (mAP) below 5 for objects classified 'small' with a fixed image resolution of 300. Resnet 101 and inception v2 outperformed the other models for detecting small objects. A tradeoff between speed and accuracy is taken into consideration while choosing the object detection model. Faster R-CNN with inception v2 recorded the slowest processing time but has the highest accuracy, while SSD-mobile recorded the fastest time with the lowest accuracy. Based on this information state of the art Faster R-CNN with Resnet 101 is selected as the object detection model. It has a medium processing time with accuracy almost equal to inception v2.

Table 1: Speed and accuracy of various object detection models

Model Name	Speed	mAP
ssd_mobilenet_v1_coco	fast	21
ssd_inception_v2_coco	fast	24
rfcn_resnet101_coco	medium	30
faster_rcnn_resnet101_coco	medium	32
faster_rcnn_inception_resnet_v2_atrous_coco	slow	37

Outline of The Thesis

The chapters of the thesis are formulated as follows:

- Chapter 2 explains the methodology and how it is implemented. It first details the technique used to acquire and create the dataset, followed by the overview of the Faster R-CNN model and how the model is trained and tested.
- Chapter 3 summarizes the results of the proposed methodology. It also details the metrics used to evaluate the processing model.
- Chapter 4 discusses the performance of the model as well as a comparative analysis of previous work.
- Chapter 5 presents the conclusions of this work and Future improvements to the project.

CHAPTER 2. METHODOLOGY

This section provides an overview of the methods to detect storm drains from google street view images. There are three significant steps followed in this approach, which are (i) create a dataset of storm drain images, (ii) Train the deep learning object detection (Faster R-CNN) model, and (iii)Process the detected image. The following sections describe the steps mentioned in detail.

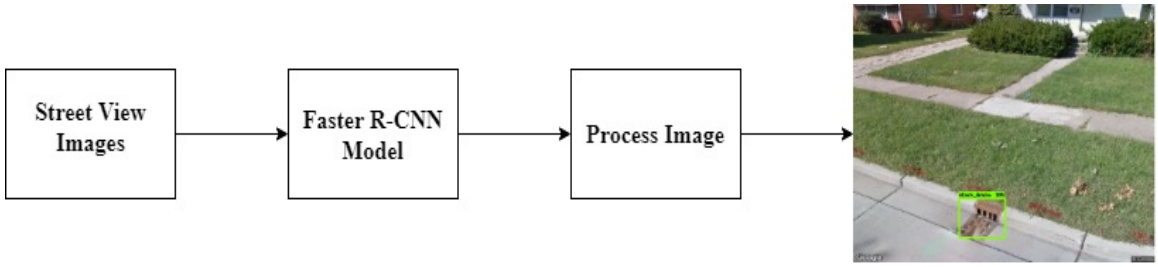

Figure 1: Overview of storm drain detection

Dataset Preparation

Data Acquisition

Street-level images are obtained from Google Street-view (GSV) and Google image search using the java GUI developed by Depwe [2]. These sources provide an extensive archive of instantly available image data. The downloaded GSV images have a resolution of 640 X 640 pixels, and if higher quality images are required, a "Google Maps API for Work" is necessary where the images can be downloaded with resolution up to 2048 x 2048. A free API Key was created to download the images. Google has a limit of 25,000 images per day for a free API key. Requesting more than 25,000 images daily for 90 consecutive days with an API key may lead to interruption or canceling of future requests.

Image perspective plays a significant role in determining the usefulness of the data downloaded. Pitch, Field of View, and Heading are the three parameters that determine the perspective of the image. These parameters are explained in the following section.

Pitch

Pitch defines the up or down angle of the camera relative to parallel with the road. Pitch values range from -90° to +90°. The value 0° meaning the angle is parallel to the road. If the angle is a positive value, for example, +45° means the camera is angled up towards the sky, and a negative value (-45°) angles the camera towards the road. Figure 3 shows an example of how the pitch parameter changes the perspective of the image. The example shown below has the pitch parameter of 0° (a), +45° (b), and -45°(c).

Figure 2: Images with different pitch values. (a)0°, (b) +45° (c) - 45°

Field of view

The field of view determines the horizontal view of the image or the zoom of the image. The value ranges from 0° to 120°. The lower the value of FOV higher the zoom level. The following figure shows the schematic of the field of view. Figure 4 shows an example of a GSV image with FOV 120° and 20°.

(a) (b)

Figure 3: Images with different fields of view. (a)20° (b)120°

Heading

Heading determines the camera rotation in degrees relative to the compass' true north. The values of heading is between 0° and 360°, with 0° being the true north and calculated clockwise with 90° being true East. The heading parameter is independent of the road direction. Figure 5 shows two images with the same heading value. Both the images are different because the road is oriented E-W in image (a) and N-S in image (b). To ensure all parts of the curb are covered for a given co-ordinate, Images were downloaded with six different heading values (N, S, NE, NW, SE, SW).

Figure 4: An example of the heading parameter to show it is independent of roadway direction. both (a) and (b) have the same heading 0°

Data Annotation

GSV images are downloaded using the application created by Depwe [2]. The images containing storm drains were manually sorted to create a dataset of 900 images, including images downloaded from google image search. A python script was written to annotate the images with a storm drain. The script allows us to draw bounding boxes on storm drains in the image, and annotation files are stored in pascal VOC format. The annotation files have the co-ordinate of the bounding box(x_max, x_min, y_max, y_min). The annotation files are used to train the Faster-RCNN ResNet-101 model for object detection.

Model Training

Overview of Faster R-CNN

Faster R-CNN is a state-of-the-art generic object detection algorithm used in many successful classification and detection problems. Faster R-CNN comprises two modules, the deep, fully convolutional network called region proposal network (RPN) and a Fast R-CNN detector. A bottleneck of the previous versions of RCNN and Fast R-CNN is the Selective search algorithm. It constitutes a significant part of the training time of the whole architecture. RPN replaces the selective search algorithm in Faster R-CNN. RPN takes an image input and generates rectangular regions within the image known as object proposal or regional proposals. The input image is resized so that the shorter size is 600px, and the longer side is not more than 1000px. Regional proposals are developed by sliding a network over the last layer of the network's convolutional layer. The network has to predict whether an object is present in the input image at its corresponding location and estimate its size for each sliding window. This is done by placing "anchors" on the input image for each site on the output feature map. Anchors are nothing but a set of reference boxes that indicates the possible objects. An anchor is centered at the current sliding window and can have multiple sizes and aspect ratios. The network moves through each pixel in these K anchors and refines the co-ordinates of the anchors to generate Region of Interest or Object proposals.

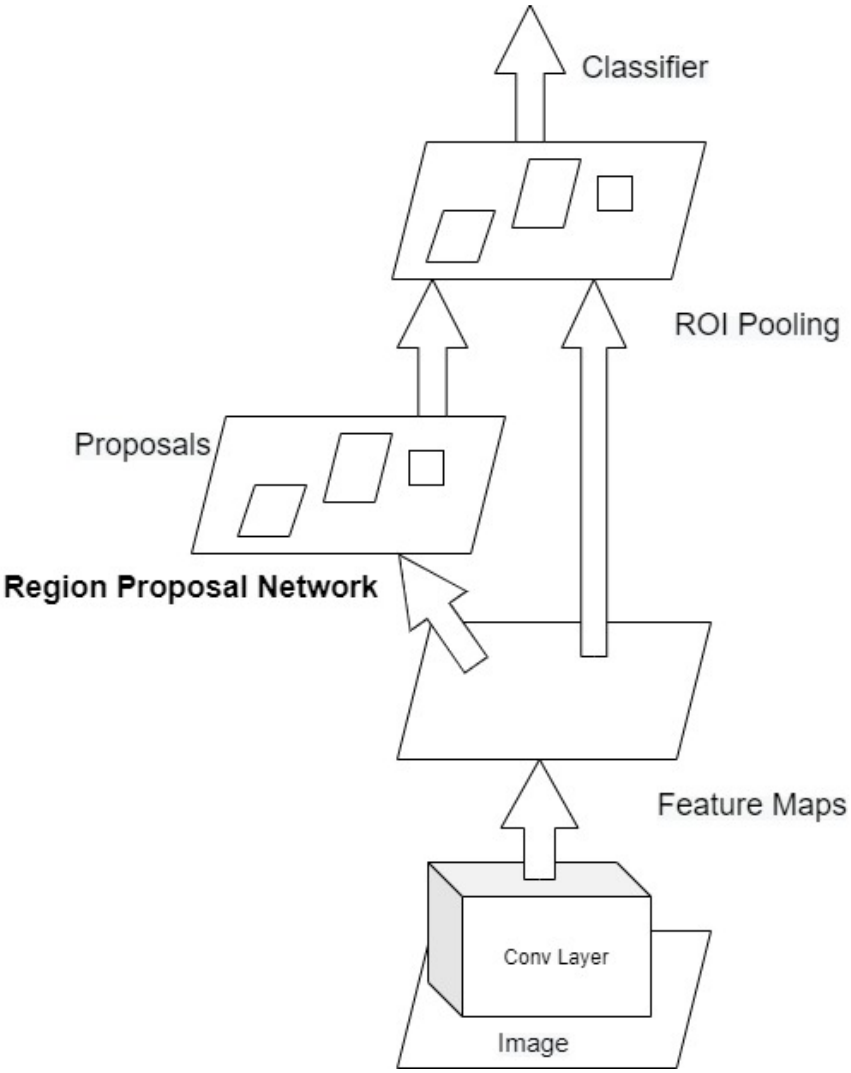

Figure 5: Faster R-CNN Architecture

An anchor is assigned a positive label when it satisfies one of the following conditions,(i) The anchor has the highest IoU (Intersection over Union, a measure of overlap) with a ground-truth box;(ii) The anchor has an IoU greater than 0.7 with any ground-truth box. A negative label is given to an anchor when IoU is less than 0.3 with all ground-truth boxes. The Remaining anchors are discarded for RPN training. RPN is trained by back-propagation and stochastic gradient descent (SGD) and follows an image-centric sampling strategy. Each Mini-Batch for training is taken from a single image, and it contains multiple positive and negative

anchors. Sampling all the anchors would bias the results towards negative samples since the negative anchors dominate the mini-batch. Alternatively,256 randomly selected anchors are used to forming the batch, which consists of positive and negative anchors in a 1:1 ratio. If there are fewer positive anchors, then the mini-batch is formed by padding additional negative anchors.

The training loss for the RPN is a multi-Task loss, and the loss function is given by equation (1)

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \quad (1)$$

i is the index of an anchor, the classification loss $L_{cls}(p_i, p_i^*)$ is the log loss over two classes. p_i is the probability of the anchor i predicted to be an object. p_i^* is the ground truth value (0 for negative and 1 for positive). The Regression loss $L_{reg}(t_i, t_i^*)$ is activated only for positive ground-truth, i.e., only when the anchor contains an object. t_i is a vector representing the four parameterized co-ordinates of the predicted bounding box, and t_i^* represents ground-truth box associated with a positive anchor. t_i and t_i^* consists of four variables each, which are $[t_X, t_Y, t_w, t_h]$ and $[t_X^*, t_Y^*, t_w^*, t_h^*]$.

Where,

$t_X = (x - x_a)/w_a$, $t_Y = (y - y_a)/h_a$, $t_w = log(^w/_{w_a})$, $t_h = log(^h/_{h_a})$

$t_X^* = (x^* - x_a)/w_a$, $t_Y^* = (y^* - y_a)/h_a$, $t_w^* = log(^{w^*}/_{w_a})$, $t_h^* = log(^{h^*}/_{h_a})$

Here x and y correspond to the (x,y) co-ordinates of the bounding box and, h and w are the height and width of the box. x_a, x* stands for the co-ordinates of the anchor box and its corresponding ground-truth bounding box. All of the anchor boxes have different regressors, and

they do not share weights. So the regression loss for an anchor *i* is applied to its corresponding regressor.

RPN and Fast R-CNN will change the convolutional layer in distinct ways if they are trained individually. Therefore, both the networks should share the convolutional layers rather than training individual ones. A four-step alternating training algorithm is devised to force the two networks to share weights. The initial step is to train the RPN by the procedure mentioned above. The network is initialized using a pre-trained ImageNet classification model and is fine-tuned for the region proposal task. The second step is to train the Fast R-CNN network with a similar ImageNet detection model. The proposals from RPN are used to train the fast R-CNN network and is fine-tuned for object detection. At this point, RPN and the fast R-CNN detector do not share a common convolutional layer. The third step is to use the tuned Fast R-CNN network to initialize the RPN, and this process is repeated. Both the network will share the same convolutional layer, and only the layers exclusive to RPN are fine-tuned. The final step is to train the Fast R-CNN detector with the new RPN. Only the layer unique to the Fast R-CNN detector is fine-tuned. This shows that Faster R-CNN detection layers have common convolutional layers.

Experiments proved that the detection algorithm using VGG RPN as a region proposal method is much faster than the selective search algorithm. Detection using VGG RPN takes 198ms compared to 1.8 seconds of selective search, which was the only bottleneck of the Fast R-CNN algorithm.

Training

The faster R-CNN model is trained with the 900 annotated files with ResNet 101 as the shared convolutional layer. ResNet-101 is 101 layers deep. ResNet 101 is pre-trained with the pets dataset. The advantage of a pre-trained model is that the trained dataset features' weights and

biases can be transferred to the storm drain dataset. It saves a considerable amount of time taken for training.

Faster R-CNN training is optimized by using stochastic gradient descent with momentum set to 0.9. The learning rate is configured to 0.0003. The model is trained for 60,000 steps with an average loss of 1.25. In the Pre-processing phase, the minimal dimension is set to 600, and the maximal dimension to 1024. The batch size is set to one. The algorithm is trained to detect only one class, "Storm Drain."

Vast.ai shared GPU is used for training. Nvidia Geforce 2080 Ti GPU is used with 11 GB of RAM. The shared instance has an Intel Xeon E5-2678 v3 CPU with 12 cores with 32 GB of RAM with a TensorFlow docker image.

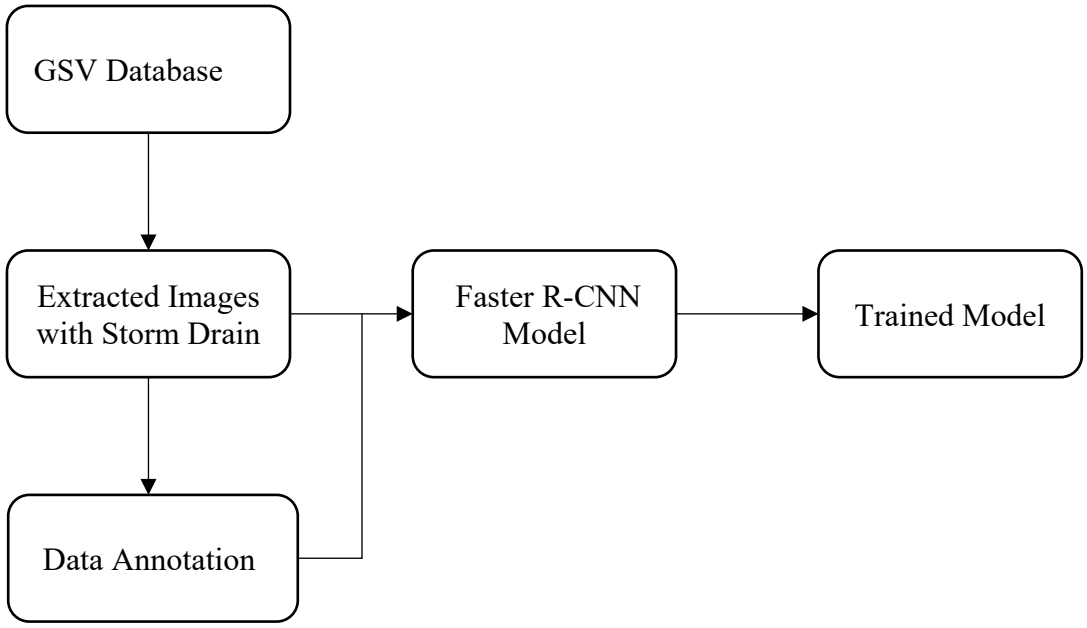

Figure 6: Dataset creation and model training

Processing Images

The final step is to process the image after it is passed through the object detection model. This step eliminates most of the false positives detected and helps in improving the accuracy of the model. Figure 8 gives an overview of the steps involved in the post-processing of the image.

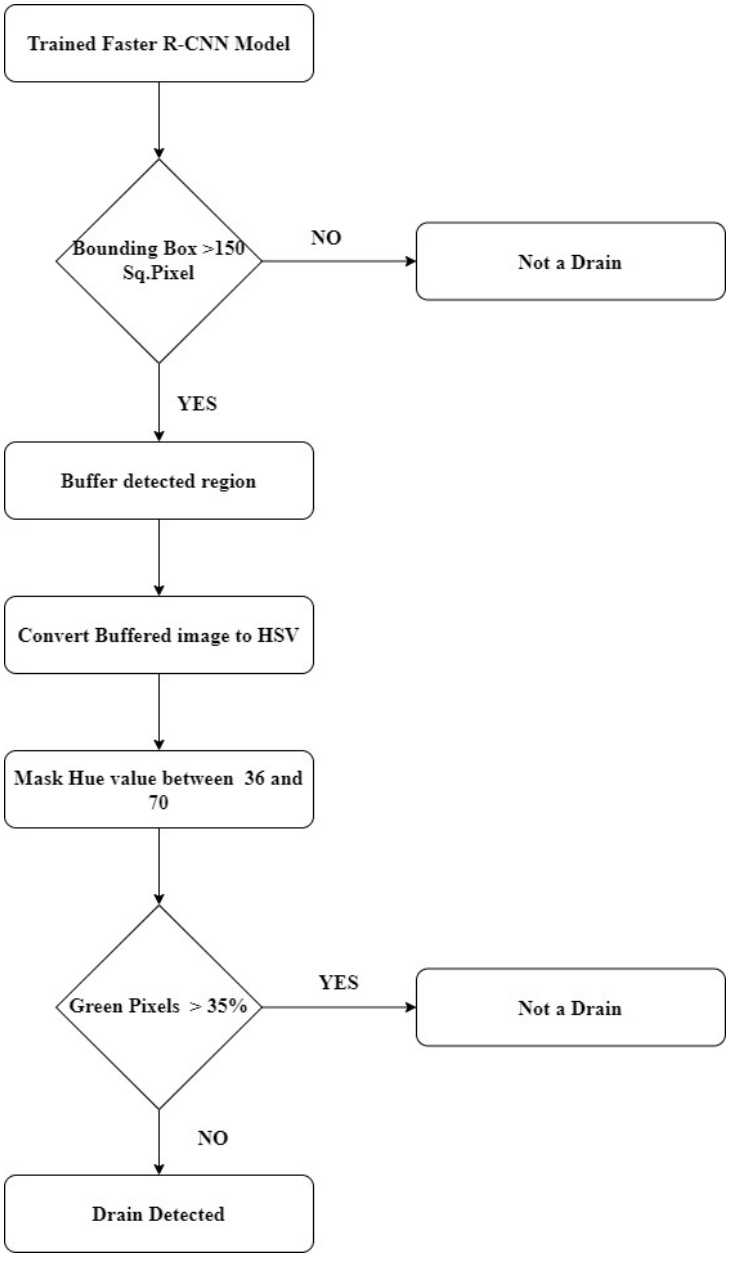

Figure 7: Overview of processing image step

Image Buffer

The object detection model creates a set of bounding-box(es) to mark the image region where it determines a drain is present. A decision was made to drop the detected bounding box if the bounding area is less than 200 square pixels. The reason for this decision is to reduce the number of false-positive detection. The bounding-boxes with an area greater than 200 $pixel^2$ are buffered with 130% of the actual size in order to apply the green filter. The reason behind buffering a greater area than the actual bounding box helps in deciding whether the detected region is more toward the road or not.

The object detection model gives the co-ordinates (x_max, x_min, y_max, y_min) of the bounding box. The image buffer is created by scaling all the four points equally and cropping the area enclosed by the scaled co-ordinates. Python Imaging Library (PIL) is used to crop the image to isolate the buffered region using the "Crop" function. The input buffered image is converted to an RGB format before cropping. This is because the faster R-CNN model computes using BGR format. In order to use the PIL library, the conversion has to be done. The crop function takes a 4-tuple defining the left, upper, right, and lower pixel co-ordinates and chops the region bounded by the co-ordinate. Figure 9 shows an example of a cropped image after the bounding area is buffered.

Figure 8: An example of an image buffer

Color Filter

Most of the storm drains located on the curbside has a road verge. Sometimes when there is a marking on the grass on the road verge or an object that resembles the drain might be detected falsely as a storm drain. To eliminate the false detection of the storm drain, a greenness filter is applied to the image's buffered region where the drain is detected. storm drains are surrounded by the road and the curb and a little portion of grass if a road verge is present. If the buffered detected region has a more green area means that the buffered image does not contain a drain.

The buffered image is converted from Red, Blue, Green (RGB) format to Hue, Saturation, and Value (HSV) format to apply the greenness filter. Opencv cvtColor function is used to convert the image from RGB to HSV. After the image is converted to HSV, pixels with Hue values between 36 and 70 are masked. The masked pixels constitute the green color in the buffered image. The number of masked pixels is calculated, and if they constitute more than 35% of the buffered image, the detection is dropped as a false positive.

Figure 9: Final detection after color filter is applied

CHAPTER 3. RESULTS

This section summarizes the results obtained through the proposed methodology. Metrics used to evaluate the obtained results are explained, and the proposed method's accuracy is reported. Training the Faster R-CNN object detection model for one epoch with 60,000 steps in vast.ai GPU took almost 8 hours. The model is tested with 1000 GSV images downloaded using java GUI. The total time to run the object detection model on the test images took 12 minutes with GPU and almost 40 minutes with a quadcore Intel Core i5 CPU.

Detection Results

The following section summarizes the results found when applying the processing algorithm after running the object detection model on the test dataset. The drain can be either grated or non-grated. The drain orientation, distance of the drain, and the image's lighting heavily impact the detection results. The impact of the parameters as mentioned earlier on the detection algorithm are discussed below with examples.

The drains in the image can be oriented either perpendicular to the roadway or angled along the road. Based on the physical distance between the drain and the GSV car's camera location when the images were recorded, the drain location is sporadic within the image. If the distance is small, the drain will on the near side of the image; else, it will be on the far side. The drains on the near side of the image appear large and have more image resolution than the drain located on the far side.

Figure 11 shows an example of the correct detection of an oriented perpendicular drain to the road, and the drain is located on the near side of the image. The image is well lit, and all the features of the drain are visible, and the object detection algorithm can detect with a 99% detection score.

Figure 10: Detection of perpendicularly oriented drains

Figure 12 and Figure 13 presents an example of a perpendicular drain in a shaded region. The drain in figure 11 is in a partially shadowed area, with most of the drain features visible and not much greener area around the drain. The algorithm was able to detect that drain. Figure 12 shows the drain is located in a fully shadowed region due to bushes and the tree in the image. The shadow masked the linear features of the drain. The drain is located at the far side of the image, and the drain's resolution is not that great. The algorithm cannot detect the drain's linear features because of the above reasons, and the drain is not detected.

Figure 11: Detection of a perpendicular drain in shadowed region

Figure 12: Perpendicular drain not detected

Unlike the perpendicularly oriented drains, angled drains are difficult to detect, and the detection depends on the angle at which the drain is oriented. Figure 14 and Figure 15 provides an example of detecting a drain that is oriented at an angle to the road. In Figure 14, the drain is closer in the image, and the drain features are clearly visible without any shadow interference. Figure 14 represents a detection of a non-grated drain located at the far side of the image. The

drain is a grated drain, but because it is on the far side, it appears as a non-grated one. Since there is no shadow noise, the drain is detected.

Figure 13: Detection of angled drain

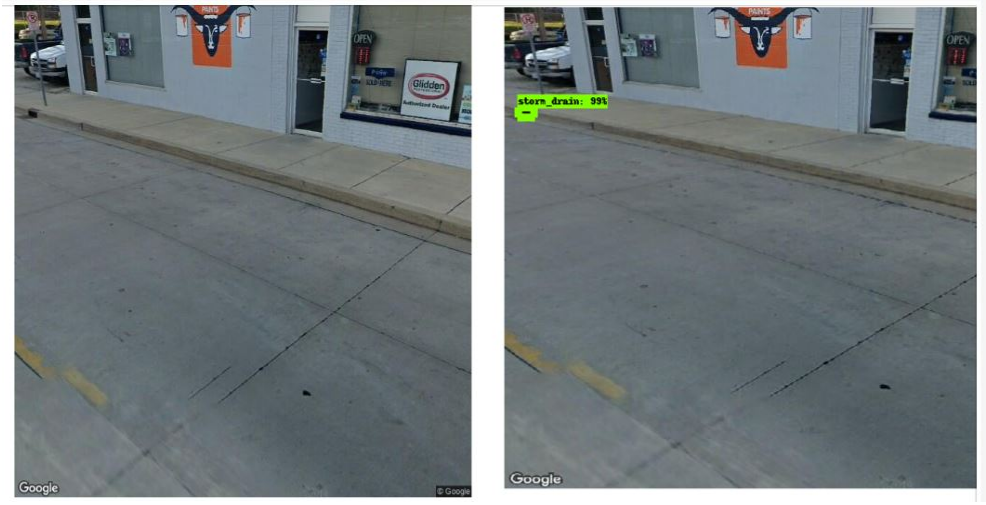

Figure 14: Detection of angled non-grated drain

Detection of angled drains becomes difficult when the angle in which the drains are oriented is decreased. Figure 16 shows an example of an angled drain which is not detected. The drain is located at the far-side, and the angle at which it is oriented is almost parallel to the road. No linear feature of the drain is not visible, and hence it is not identified.

Figure 15: Angled drain not detected

Some environmental features resulted in false-positive detection. Heaps of dried leaf and shadowed dirt made patterns that resemble a drain. These patterns were classified as storm drains in some instances. Figure 17 shows an example of a false positive classification due to dried leaves. The color of the heaped leaves are darker and are grouped in a manner near the curb that closely resembles a grated drain.

Figure 16: False positive detection due to ground leaf foliage

Apart from the color filter, which is used to reduce the false positives as explained in the methodology section, a pre-processing step of detecting the curb on the road using hough-transform and canny edge filters were tried. The idea was to detect the curbs and apply the object-detection algorithm, only the detected curb region. This proved to be futile because the road's orientation was not consistent in the downloaded images. There were very little to no distinguishable features between the road and the curb, and the shadow and image quality hampered the edge detection techniques.

Evaluation Metrics

The object detection model's performance is evaluated using four parameters: Accuracy, Precision, Recall, and F1 Score.

Accuracy is the ratio of correctly detected observations to the total observations. The following formula calculates accuracy.

$$Accuracy = \frac{TP}{TP + FP + FN}$$

Precision is the ratio of detected true positive observations of the total predicted positive observations. The high value of precision relates to a low false-positive rate.

$$Precision = \frac{TP}{TP + FP}$$

The recall is the ratio of True prediction of drains to the total number of actual drains in the test dataset.

$$Recall = \frac{TP}{TP + FN}$$

F1 score is the weighted mean of Precision and Recall.

$$F1\ Score = 2 * \frac{Recall * Precision}{Recall + Precission}$$

Where TP is True positive, which indicates the total number of drains successfully detected, TN is True Negative, which gives the number of not-drains successfully detected, FP is False Positive, which gives the number of non-drains objects detected as drains, and FN is False Negative which indicates the number of drains not detected by the algorithm.

A test dataset with 1000 images having 600 instances of storm drain was used to evaluate the Faster R-CNN model and compared the performance with the SSD mobilenet v2. SSD mobilenet was trained for 80000 steps with an average training loss of 1.65. The Faster R-CNN model could detect 480 cases of the storm correctly with 84 False Negatives and 36 False Positives. Whereas in SSD mobilent, the number of false positives is reduced to 25, but the number of true detections also took a hit. The number of true detections was reduced to 400 out of 600 drain instances. Table 1 gives the calculated values for the evaluation metrics.

Table 2: Evaluation Metrics.

Metric	Faster R-CNN	SSD Mobilenet
Accuracy	80%	66.67%
Precision	93.03%	94.11%
Recall	85.10%	69.5%
F1 score	88.89%	79.95%

CHAPTER 4. DISCUSSIONS

This Chapter discusses the results presented in the previous Chapter. The evaluation of the metrics calculated for the Object detection model is explained. Comparing the results obtained by the proposed methodology with the earlier work on storm drain detection is discussed.

Performance Evaluation

Table 1 gives the calculated values of the evaluation metrics. The Faster R-CNN algorithm has an accuracy of 80%, which is much higher than mobilenet's 66.67%. The algorithm's accuracy is greatly affected by the image resolution and the noise (shadows) in the image. Image orientation influenced the accuracy, but it did not affect the classification of most of the drains. Also, if a drain is obstructed due to objects like a car or a heap of leaves, the drain is not identified.

The Faster R-CNN algorithm has a precision value of 93.03, while the SSD mobilenet has a slightly higher precision of 94%. The high value indicates that when the algorithm classifies an object as a storm drain, it is highly probable that the detected object is a drain. This proves that the algorithm has a high ability to detect drains correctly. The percentage of the non-drain objects classified as the drain is meager.

The Recall value determines the algorithm's ability to detect multiple instances of the drains in the image. Faster R-CNN has a recall value of 85.10% compared to SSD mobilenet's 69.5%. This is due to the high number of False Negatives(84 for Faster R-CNN and 175 for SSD Mobilenet). Most of the false negatives are expected due to shadow in the image and drains located on the far side.

F1 score determines the robustness of the algorithm. The harmonic average of precision and recall determines the algorithm's ability to detect all instances of the drain with no False Positives.F1 score has a value of 88, proving that the Faster R-CNN algorithm is robust in

extracting drains compared to an F1 score of 79.95 with SSD mobilenet. The results obtained proved that the Faster R-CNN model outperformed the SSD mobilenet. Even though the precision of SSD is slightly higher due to less number of false positives, but a large number of false negatives affected the overall performance of the model.

Comparative Analysis

Earlier work on detecting storm drains from GSV follows the Computer Vision approach. One of the significant defects of that approach was the occurrence of a considerable number of False Positives. The False Positives are due to noise in the image caused by shadows due to tall trees and patterns made by leaf foliage and house fencing. The accuracy of the processing model using the Computer Vision method took a hit because of the False Positives. The deep proposed deep learning model has reduced the occurrence of false positives to a considerable number. Figure 18 compares the result of the Computer Vision model (left) and the proposed deep learning model (right). The Computer Vision based algorithm detected the patterns formed by the tree's shadow on the lawn as drains. The proposed model outperforms the Computer vision model with zero false positives on the example image.

Figure 17: Comparison of computer vision model and deep learning model.

Another significant improvement with the proposed model is the detection of non-grated drains. The nearness filter in the OpenCV method inhibits the detection of non-grated drains. Although there were not enough images of non-grated drains for the training dataset, their detection is less reliable than grated drains. The Significant thing to be noted here is that most of the false positive detections were of different types of drain outlets on the roadway and some markings on the lawn grass. Figure 19 shows an example of a drain outlet between the curbs identified as a storm drain.

Figure 18: False positive detection of a drain outlet

CHAPTER 5. CONCLUSION

Summary

Stormwater management is designed to protect and conserve the environment. These systems are highly critical for both urban and rural infrastructure design. Street-level storm drains forms the backbone of this vital infrastructure design. Knowledge of the storm drains' location and functionality helps in understanding and improving the infrastructure of the runoff and sewage water management system. Because of the frameworks' multifaceted nature, the recurrence of updates and the information's precision can be issues.

Earlier work on storm drain detection found that Google street view imagery is a practical method for detecting and obtaining storm drain information. Computer Vision image processing techniques were used to process the street-level images to identify storm drains. Accuracy in detecting storm drains took a hit due to the high frequency of false classification of drains. This called for a robust processing algorithm to identify the drains with minimal false detection.

A deep learning approach using the Faster R-CNN Resnet-101 model was developed to process the street view images to detect the storm drains. A storm drain dataset is created with 1000 images to train the model. With the deep learning object detection model, the accuracy of detecting storm drain is improved by reducing false detection. The false detection was reduced to just 3.6% in over 1000 tested images.

Future Work

The next steps on this project would be improving the drain dataset with different types of storm drains and inlets and improving the algorithm in detecting various different classes of drains, porting the object detection code to an online portal called "openstormdrain.org." This will help the user to get all the information regarding the location of the storm drain, which was not accessible earlier. This will also help in updating the storm drain database and will help in future improvement of the stormwater management system.

The data set has been provided publicly. This will help future researchers and Engineers in the urban infrastructure domain because this dataset is difficult to obtain. Various other Deep Learning algorithms can be proposed and tested to produce a more specific network for Drain Detection.

REFERENCES

[1] EPA, "Report to Congress: Impacts and Control of CSOs and SSOs," EPA, Washington,DC, 2004.

[2] E. DEPWE, "Extracting curbside Storm Drain locations from street-level images," 2015.

[3] F. Zhang, L. Wu, D. Zhu and Y. Liu, "Social sensing from street-level imagery: A case study in learning spatio-temporal urban mobility patterns," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 153, pp. 48-53, 2019.

[4] T. Gebru, J. Krause, Y. Wang, D. Chen, J. Deng, E. L. Aiden and L. Fei-Fei, *Using deep learning and Google street view to estimate the demographic makeup of the US,* [Online] Available: https://arxiv.org/abs/1702.06683., 2017.

[5] Y. Lu, J. Lu, S. Zhang and P. Hall, "Traffic signal detection and classification in street views using an attention model," *Computational Visual Media,* vol. 4, no. 3, pp. 253-266, 2018.

[6] A. G. Rundle, M. D. Bader, C. A. Richards, K. M. Neckerman and J. O. Teitler, "Using google street view to audit neighborhood environments," *American Journal of Preventive Medicine,* vol. 40, no. 1, pp. 94-100, 2011.

[7] J. P. e. al, "Detection of Manhole Covers in High-Resolution Aerial Images of Urban Areas by Combining Two Methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. 9, no. 5, pp. 1802-1807, 2016.

[8] A. Campbell, A. Both and Q. (. Sun, "Detecting and mapping traffic signs from Google Street View images using deep learning and GIS," *Computers, Environment and Urban Systems,* vol. 77, 2019.

[9] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 38, no. 1, pp. 142-158, 2016.

[10] R. Girshick, "Fast R-CNN," in *International Conference on Computer Vision (ICCV)*, 2015.

[11] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, no. 6, pp. 1137-1149, 2017.

[12] I. Google, "Google Developers: Google Street View API," [Online]. Available: https://developers.google.com/maps/documentation/streetview/overview. [Accessed January 2019].

[13] M. D. o. Transportation, "Chapter 8 Storm Drainage Systems," in *Drainage Manual*, 2000.

[14] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar and K. Ouni, "Car Detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3," in *International Conference on Unmanned Vehicle Systems (UVS)*, Muscat,Oman, 2019.

APPENDIX [TRAINING DATASET]

The training dataset with the annotated files used for training is available in the following link. https://iastate.box.com/s/22eiam154lb6ffwyb8kt10onosc6de8f